Intel® Xeon™ Processor MP Specification Update

Release Date: July 2002

Order #: 290741-005

The Intel® Xeon™ Processor MP may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

The Intel® Xeon™ Processor MP may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

The Specification Update should be publicly available following the last shipment date for a period of time equal to the specific product's warranty period. Hardcopy Specification Updates will be available for one (1) year following End of Life (EOL). Web access will be available for three (3) years following EOL.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's website at http://www.intel.com.

CONTENTS

REVISION HISTORY

Date of Revision	Version	Description
March 2002	-001	First Public Release.
April 2002	-002	Addition of “Mixed Steppings in MP Systems” section. Added 5 new Documentation Changes. Added erratum O39.
May 2002	-003	Added PWRGOOD Specification change Added Errata 040. Updated Errata 012, O29, Added Specification Changes O1 and O2. Added Documentation Changes O1-O3.
June 2002	-004	Added Errata O41 and O42. Added new Documentation Changes O1- O2
July 2002	-005	Added new Erratum O43 Added new Documentation Changes O3-O12.

PREFACE

This document is an update to the specifications contained in the following documents:

- *Intel® Xeon™ Processor MP* datasheet (Order Number 29074001)
- *Intel Architecture Software Developer's Manual*, *Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools. It contains Errata, Documentation Changes, Specification Clarifications and Specification Changes.

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the behavior of the Intel® Xeon™ Processor MP to deviate from published specifications. Hardware and software designed to be used with any given processor must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications. These changes will be incorporated in the next release of the specifications.

GENERAL INFORMATION

INTEL® XEON™ PROCESSOR MP MARKINGS (603-pin INT-mPGA)

Figure 1. Top Side Processor Marking – Production Part

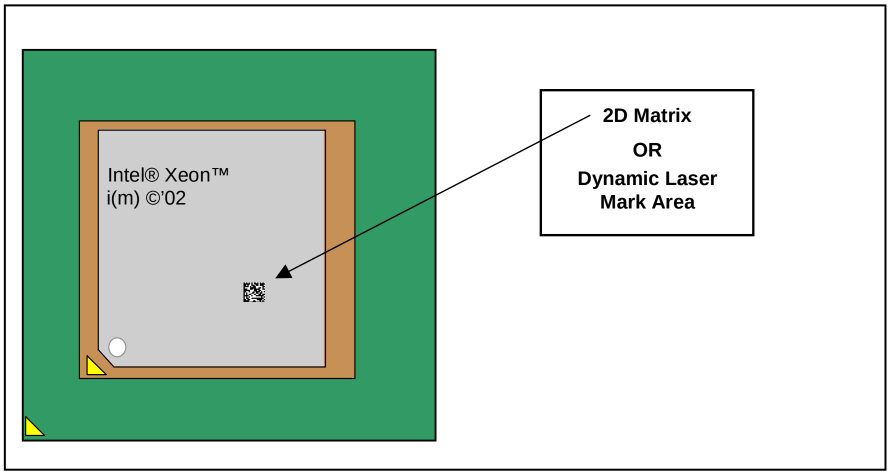
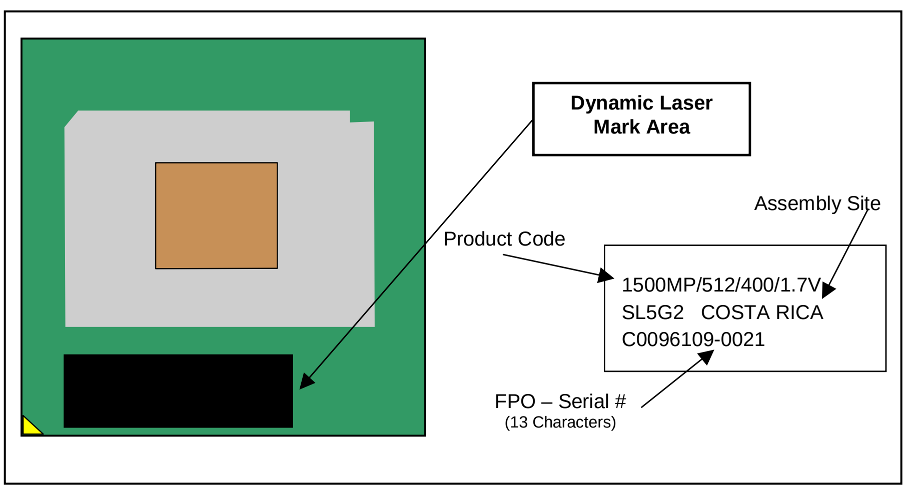


Figure 2. Bottom Side Processor Marking



IDENTIFICATION INFORMATION

The Intel® Xeon™ Processor MP can be identified by the following values:

Family[1]	**Model**[2]	**Brand ID**[3]
1111	0001	0000 1011

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after RESET, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the generation field of the Device ID register accessible through Boundary Scan.
2. The Model corresponds to bits [7:4] of the EDX register after RESET, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the model field of the Device ID register accessible through Boundary Scan.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a 1 in the EAX register.

Cache and TLB descriptor parameters are provided in the EAX, EBX, ECX and EDX registers after the CPUID instruction is executed with a 2 in the EAX register. Please refer to the *Intel Processor Identification and the CPUID Instruction Application Note* (AP-485) for further information on the CPUID instruction.

MIXED STEPPINGS IN MP SYSTEMS

Intel Corporation fully supports mixed steppings of Intel® Xeon™ processors MP. The following list and processor matrix describes the requirements to support mixed steppings:

- Mixed steppings are only supported with processors that have identical family numbers as indicated by the CPUID instruction. The Intel® Xeon™ processor is available with two different Model numbers as indicated by the CPUID. Please refer to the "MP Platform Population Matrix for the Intel® Xeon™ Processor" for details regarding inclusion of processors with mixed CPUID/Core steppings.
- While Intel has done nothing to specifically prevent processors operating at differing frequencies from functioning within a multiprocessor system, there may be uncharacterized errata that exist in such configurations. Intel does not support such configurations. In mixed stepping systems, all processors must operate at identical frequencies (i.e., the highest frequency rating commonly supported by all processors).
- While there are no known issues associated with the mixing of processors with differing cache sizes in a multiprocessor system, and Intel has done nothing to specifically prevent such system configurations from operating, Intel does not support such configurations since there may be uncharacterized errata that exist. In mixed stepping systems, all processors must be of the same cache size.
- While Intel believes that certain customers may wish to perform validation of system configurations with mixed frequency or cache sizes, and that those efforts are an acceptable option to our customers, customers would be fully responsible for the validation of such configurations.
- Intel requires that the proper microcode update be loaded on each processor operating in a multiprocessor system. Any processor that does not have the proper microcode update loaded is considered by Intel to be operating out of specification.
- The workarounds identified in this and following specification updates must be properly applied to each processor in the system. Certain errata are specific to the multiprocessor environment and are identified in the *Mixed Stepping Processor Matrix* found at the end of this section. Errata for all processor steppings will affect system performance if not properly worked around. Also see the Intel® Xeon™ Processor MP Identification and Package Information section for additional details on which processors are affected by specific errata.
- In mixed stepping systems, the processor with the lowest feature-set, as determined by the CPUID Feature Bytes, must be the Bootstrap Processor (BSP). In the event of a tie in feature-set, the tie should be resolved by selecting the BSP as the processor with the lowest stepping as determined by the CPUID instruction.

In the following processor matrix, **"NI"** indicates that there are currently no known issues associated with mixing these steppings. A number indicates that a known issue has been identified as listed in the table following the matrix. **"X"** indicates the processors cannot be mixed. A multiple processor system using mixed processor steppings must assure that errata are addressed appropriately for each processor.

MP Platform Population Matrix for the Intel® Xeon™ Processor MP	
CPUID/Core Stepping	**0F11h/C0**
0F11h/C0	**NI**

Intel® Xeon™ Processor MP Identification Information

S-Spec	Core Stepping	Processor Signature	Speed		L2 Cache Size	L3 Cache Size	Hyper-Threading Technology	Processor Interposer Revision	Package And Revision[2]	S-Spec Notes
			Core (GHz)	Data Bus (MHz)						
SL5G8	C0	0F11h	1.60	400	256KB	1 MB	Yes	B0	42.5 mm OLGA rev 1.0	1
SL5S4	C0	0F11h	1.60	400	256KB	1 MB	Yes	B0	42.5 mm OLGA rev 1.0	1, 3
SL5FZ	C0	0F11h	1.40	400	256KB	512KB	Yes	B0	42.5 mm OLGA rev 1.0	1
SL5RV	C0	0F11h	1.40	400	256KB	512KB	Yes	B0	42.5 mm OLGA rev 1.0	1, 3
SL5G2	C0	0F11h	1.50	400	256KB	512KB	Yes	B0	42.5 mm OLGA rev 1.0	1
SL5RW	C0	0F11h	1.50	400	256KB	512KB	Yes	B0	42.5 mm OLGA rev 1.0	1, 3

NOTES:

1. These parts require the inputs from A20M#, IGNNE#, LINT[1]/NMI and LINT[0]/INTR pins during RESET to set the correct core to bus frequency ratio.
2. The **Intel® Xeon™ Processor MP listed here is installed onto a micro pin grid array (mPGA) interposer. The overall processor package is called INT-mPGA.**
3. This part is an Intel boxed processor.

SUMMARY OF CHANGES

The following table indicates the Errata, Documentation Changes, Specification Clarifications, or Specification Changes that apply to the Intel® Xeon™ Processor MP. Intel intends to fix some of the errata in a future stepping of the component, and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notation:

CODES USED IN SUMMARY TABLE

X:	Specification Change, Erratum, Specification Clarification, or Documentation Change applies to the given processor stepping.
(No mark) or (blank box):	This item is fixed in or does not apply to the given stepping.
Doc:	Intel intends to update the appropriate documentation in a future revision.
PlanFix:	This erratum may be fixed in a future stepping of the product.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.
Shaded:	This item is either new or modified from the previous version of the document.

Each Specification Update item will be prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Intel® Pentium® II processor

B = Mobile Intel® Pentium® II processor

C = Intel® Celeron® processor

D = Intel® Pentium® II Xeon™ processor

E = Intel® Pentium® III processor

G = Intel® Pentium® III Xeon™ processor

H = Mobile Intel® Celeron® processor at 466 MHz, 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz

K = Mobile Intel® Pentium® III Processor - M

M = Mobile Intel® Celeron® processor

N = Intel® Pentium® 4 processor

O = Intel® Xeon™ processor MP

P = Intel® Xeon™ processor and Intel® Xeon™ processor with 512K L2 Cache

T = Mobile Intel® Pentium® 4 processor

V = Mobile Intel® Celeron® processor on .13 Micron Process in Micro-FCPGA Package

The Specification Updates for the Pentium® processor, Pentium® Pro processor, and other Intel products do not use this convention.

Summary of Errata

NO.	C0 Core Step	PLANS	ERRATA
O1	X	No Fix	UC Code in same line as WriteBack (WB) data may lead to data corruption
O2	X	No Fix	Transaction is not retried after BINIT#
O3	X	No Fix	Invalid opcode 0FFFH requires a ModRM byte
O4	X	No Fix	FSW may not be completely restored after page fault on FRSTOR or FLDENV Instructions
O5	X	No Fix	Shutdown and IERR# may result due to MC on Hyper-Threading Technology enabled processor
O6	X	No Fix	4M mode B pages and 2M mode C pages in no-fill mode are set to uncacheable
O7	X	No Fix	Processor may hang due to Speculative Page Walks to Non-Existent System Memory
O8	X	No Fix	Writing a performance counter may result in an incorrect counter value
O9	X	No Fix	Performance counter may contain incorrect value after being stopped
O10	X	No Fix	Memory type of the Load Lock is different from its corresponding Store Unlock
O11	X	No Fix	Machine Check Architecture error reporting and recovery may not work as expected
O12	X	No Fix	Debug Mechanisms may not function as expected
O13	X	No Fix	Processor may timeout waiting for a device to respond after 0.67 seconds
O14	X	No Fix	Cascading of performance counters does not work correctly when forced overflow is enabled
O15	X	No Fix	EMON event counting of x87 loads may not work as expected
O16	X	No Fix	Simultaneous code breakpoint and uncorrectable error results in a processor hang
O17	X	No Fix	Software controlled clock modulation using a 12.5% or 25% duty cycle may cause the processor to hang
O18	X	No Fix	Processor samples bus frequency power-on configuration pins at the assertion of PWRGOOD
O19	X	No Fix	PAT index MSB may be calculated incorrectly
O20	X	No Fix	System bus interrupt messages without data which receive a HardFailure response may hang the processor
O21	X	No Fix	Bus Invalidate Line requests that return unexpected data may result in L1 cache corruption
O22	X	No Fix	Processor flags #PF instead of #AC on an unlocked CMPXCHG8B instruction
O23	X	No Fix	Incorrect data may be returned when page tables are located in Write Combining (WC) memory
O24	X	No Fix	Multi-processor boot protocol may not complete with an IOQ depth of one
O25	X	No Fix	Write Combining (WC) load may result in an unintended address on system bus
O26	X	No Fix	Processor issues inconsistent transaction size attributes for locked operation
O27	X	No Fix	Multiple accesses to the same S-state L2 cache line and ECC error combination may result in loss of cache coherence
O28	X	No Fix	IA32_MC0_ADDR and IA32_MC0_MISC registers will contain invalid or stale data following a Data, Address, or Response Parity Error
O29	X	No Fix	Instruction Pointer stored on stack may become invalid
O30	X	No Fix	When the processor is in the System Management Mode (SMM), Debug Registers may be fully writeable

Summary of Errata

NO.	C0 Core Step	PLANS	ERRATA
O31	X	No Fix	Associated counting logic must be configured when using Event Selection Control (ESCR) MSR
O32	X	No Fix	Livelock may occur when bus parking is disabled
O33	X	No Fix	CPUID Function 2 may return incorrect cache size information
O34	X	No Fix	CR2 may be incorrect or an incorrect page fault error code may be pushed onto stack after execution of an LSS instruction
O35	X	No Fix	Hyper-Threading Technology enabled processors may hang in the presence of extensive self-modifying code
O36	X	No Fix	Global bit incorrectly set for secondary logical processors in ITLB
O37	X	No Fix	Hardware Prefetcher may cause stale data to be loaded into the processor caches
O38	X	No Fix	System may hang if a fatal cache error causes Bus Write Line (BWL) transaction to occur to the same cache line address as an outstanding Bus Read Line (BRL) or Bus Read-Invalidate Line (BRIL)
O39	X	No Fix	Re-mapping the APIC Base Address to a value less than or equal to 0xDC001000 may cause IO and Special Cycle Failure
O40	X	Fix	Erroneous Machine Check Error Reported
O41	X	No Fix	Processor does not Flag #GP on no-zero write to certain MSRs
O42	X	No Fix	Counting both L2 and L3 Cache reference events may result in undercount
O43	X	No Fix	Simultaneous assertion of A20M# and INIT# may result in incorrect data fetch

Summary of Documentation Changes

NO.	C0 Core Stepping	PLANS	Documentation Changes
O1	X	Doc	SSE and SSE2 instructions opcodes
O2	X	Doc	Executing the SSE2 variant on a non-SSE2 capable processor
O3	X	Doc	ISR Must Re-enable CCCR After Each PEBS Overflow
O4	X	Doc	Sequence to Programming Performance Counters
O5	X	Doc	0x2B MSR Definition
O6	X	Doc	ESI and EDI Alignment For Fast String Moves
O7	X	Doc	BUS_UTILIZATION_DUE_TO_PROCESSOR_ACTIVITY Event Number Correction
O8	X	Doc	Complement flag, bit 19
O9	X	Doc	BSF and BSR Incorrectly Documented in Vol 2 Appendix B
O10	X	Doc	Programming Performance MSRs (MSR_IQ_COUNTER)
O11	X	Doc	Tagging mechanism for Replay_Event
O12	X	Doc	Update to Table B-2, MSRs in the P6 Family Processors

Summary of Specification Clarifications

NO.	Plans	SPECIFICATION CLARIFICATIONS
		There are no Specification Clarifications

Summary of Specification Changes

NO.	Plans	SPECIFICATION CHANGES
		There are no Specification Clarifications

ERRATA

O1. UC Code in Same Line as WriteBack (WB) Data May Lead to Data Corruption

Problem: This erratum occurs when both code (being accessed as UC or WC) and data (being accessed as WB) are placed in the same cache line. The UC fetch will cause the processor to self-snoop and generate an implicit writeback. The data supplied by this implicit writeback may be corrupted due to the way the processor is currently handling self-modifying code.

Implication: UC code located in the same cache line as WB data may lead to data corruption.

Workarounds: UC or WC code should not be located in the same 64-byte cache line as any location that is being stored to with WB data.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O2. Transaction is not Retried after BINIT#

Problem: If the first transaction of a locked sequence receives a HITM# and DEFER# during the snoop phase it should be retried and the locked sequence restarted. However, if BINIT# is also asserted during this transaction, the transaction will not be retried.

Implication: When this erratum occurs, locked transactions will not be retried.

Workaround: None identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O3. Invalid Opcode 0FFFh Requires a ModRM Byte

Problem: Some invalid opcodes require a ModRM byte and other following bytes, while others do not. The invalid opcode 0FFFh did not require a ModRM in previous generation microprocessors such as Pentium® II or Pentium III processors, but it is required in the Intel® Xeon™ Processor MP.

Implication: The use of an invalid opcode 0FFFh without the ModRM byte may result in a page or limit fault on the Intel® Xeon™ Processor MP.

Workaround: To avoid this erratum use ModRM byte with invalid 0FFFh opcode.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O4. FSW May not be Completely Restored after Page Fault on FRSTOR or FLDENV Instructions

Problem: If the FPU operating environment or FPU state (operating environment and register stack) being loaded by an FLDENV or FRSTOR instruction wraps around a 64Kbyte or 4Gbyte boundary and a page fault (#PF) or segment limit fault (#GP or #SS) occurs on the instruction near the wrap boundary, the upper byte of the FPU status word (FSW) might not be restored. If the fault handler does not restart program execution at the faulting instruction, stale data may exist in the FSW.

Implication: When this erratum occurs, stale data will exist in the FSW.

Workaround: Ensure that the FPU operating environment and FPU state do not cross 64Kbyte or 4Gbyte boundaries. Alternately, ensure that the page fault handler restarts program execution at the faulting instruction after correcting the paging problem.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O5. Shutdown and IERR# May Result Due to a Machine Check Exception on a Hyper-Threading Technology Enabled Processor

Problem: When a Machine Check Exception (MCE) occurs due to an internal error, both logical processors on a Hyper-Threading Technology enabled processor normally vector to the MCE handler. However, if one of the logical processors is in the “Wait for SIPI” state, that logical processor will not have a MCE handler and will shut down and assert IERR#.

Implication: A processor with a logical processor in the “Wait for SIPI” state will shut down when an MCE occurs on the other thread.

Workaround: None identified at this time.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O6. When in No-Fill Mode (CR0.CD=1) the Memory Type of Large (PSE-4M and PAE-2M) Pages are Wrongly Forced to Uncacheable

Problem: When the processor is operating in No-Fill Mode (CR0.CD=1), the page miss hardware incorrectly forces the memory type of large (PSE-4M and PAE-2M) pages to UC memory type regardless of the MTRR settings. By forcing the memory type of these pages to UC, load operations, which should hit valid data in the L1 cache, are forced to load the data from system memory. Some applications will lose the performance advantage associated with the caching permitted by other memory types.

Implication: This erratum may result in some performance degradation when using no-fill mode with large pages.

Workaround: None identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O7. Processor May Hang Due to Speculative Page Walks to Non-Existent System Memory

Problem: A load operation issued speculatively by the processor that misses the Data Translation Lookaside Buffer (DTLB) results in a page-walk. A branch instruction older than the load retires so that this load operation is now in the mispredicted branch path. Due to an internal boundary condition, in some instances the load is not cancelled before the page walk is issued.

The Page Miss Handler (PMH) starts a speculative page-walk for the Load and issues a cacheable load of the Page Directory Entry (PDE). This PDE loads returns data that points to a page table entry in uncacheable (UC) memory. The PMH issues the PTE Load to UC space, which is issued on the system bus. No response comes back for this load PTE operation since the address is pointing to system memory that does not exist.

This load to non-existent system memory causes the processor to hang because other bus requests are queued up behind this UC PTE load which never gets a response. If the load was accessing valid system memory, the speculative page-walk would successfully complete and the processor would continue to make forward progress.

The boundary conditions to generate this erratum are more likely to occur with Hyper-Threading Technology enabled but may also occur with Hyper-Threading Technology disabled.

Implication: Processor may hang due to speculative page walks to non-existent system memory.

Workaround: Page directories and page tables in UC memory space must point to system memory that exists.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O8. Writing a Performance Counter May Result in an Incorrect Counter Value

Problem: Accessing a performance counter also enables the counter input so that writing one half of the counter can cause the other half to increment. When a performance counter is written and the event counter for the event being monitored is non-zero, the performance counter will be incremented by the value on that event counter. Because the upper eight bits of the performance counter are not written at the same time as the lower 32 bits, the increment due to the non-zero event counter may cause a carry to the upper bits such that the performance counter contains a value higher than what was written. The worst-case error caused by this can be about 4 billion counts.

Implication: When this erratum occurs, the performance counter will contain a different value from that which was written.

Workaround: If the performance counter is set to select a null event and the CCCR for that counter has its compare bit set to zero, before the performance counter is written, this erratum will not occur.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O9. Performance Counter May Contain Incorrect Value After Being Stopped

Problem: If a performance counter is stopped on the precise internal clock cycle where the intermediate carry from the lower 32 bits of the counter to the upper eight bits occurs, the intermediate carry is lost.

Implication: When this erratum occurs, the performance counter may contain a value about 4 billion (2^{32}) less than it should.

Workaround: Since this erratum does not occur if the performance counters are read when running, a possible workaround is to read the counter before stopping it.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O10. Memory Type of the Load Lock Different from its Corresponding Store Unlock

Problem: The Intel Xeon Processor MP employs a use-once protocol to ensure that a processor in a multi-processor system may access data that is loaded into its cache on a Read-for-Ownership operation at least once before it is snooped out by another processor. This protocol is necessary to avoid an MP livelock scenario where no processor in the system can gain ownership of a line and modify it before that data is snooped out by another processor. In the case of this erratum, the use-once protocol incorrectly activates for split load lock instructions. A load lock operation accesses data that splits across a page boundary with both pages of WB memory type. The use-once protocol activates and the memory type for the split halves get forced to UC. Since use-once does not apply to stores, the store unlock instructions go out as WB memory type. The full sequence on the Bus is: locked partial read (UC), partial read (UC), partial write (WB), locked partial write (WB). The Use-once protocol should not be applied to Load locks

Implication: When this erratum occurs, the memory type of the load lock will be different than the memory type of the store unlock operation. This behavior (Load Locks and Store Unlocks having different memory types) does not however introduce any functional failures such as system hangs or memory corruption.

Workaround: None identified at this time.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O11. Machine Check Architecture Error Reporting and Recovery May Not Work as Expected

Problem: When the processor detects errors it should attempt to report and/or recover from the error. In the situations described below, the processor does not report and/or recover from the error(s) as intended.

- When a transaction is deferred during the snoop phase and subsequently receives a Hard Failure response, the transaction should be removed from the bus queue so that the processor may proceed. Instead, the transaction is not properly removed from the bus queue, the bus queue is blocked, and the processor will hang.
- When a hardware prefetch results in an uncorrectable tag error in the L2 cache, MC0_STATUS.UNCOR and MC0_STATUS.PCC are set but no Machine Check Exception (MCE) is signaled. No data loss or corruption occurs because the data being prefetched has not been used. If the data location with the uncorrectable tag error is subsequently accessed, an MCE will occur. However, upon this MCE, or any other subsequent MCE, the information for that error will not be logged because MC0_STATUS.UNCOR has already been set and the MCA status registers will not contain information about the error which caused the MCE assertion but instead will contain information about the prefetch error event.
- When the reporting of errors is disabled for Machine Check Architecture (MCA) Bank 2 by setting all MC2_CTL register bits to 0, uncorrectable errors should be logged in the IA32_MC2_STATUS register but no machine-check exception should be generated. Uncorrectable loads on bank 2, which would normally be logged in the IA32_MC2_STATUS register, are not logged.
- When one half of a 64 byte instruction fetch from the L2 cache has an uncorrectable error and the other 32 byte half of the same fetch from the L2 cache has a correctable error, the processor will attempt to correct the correctable error but cannot proceed due to the uncorrectable error. When this occurs the processor will hang.
- When an L1 cache parity error occurs, the cache controller logic should write the physical address of the data memory location that produced that error into the IA32_MC1_ADDR REGISTER (MC1_ADDR). In some instances of a parity error on a load operation that hits the L1 cache, however, the cache controller logic may write the physical address from a subsequent load or store operation into the IA32_MC1_ADDR register.
- The local xAPIC has an Error Status Register, which records all errors it detects. Bit 6 of this register, the Receive Illegal Vector bit, is set when the local xAPIC detects an illegal vector in a message that it received. When an illegal vector error is received on the same internal clock that the error status register is being written due to a previous error, bit 6 does not get set and illegal vector errors are not flagged.
- When an error exists in the tag field of a cache line such that a request for ownership (RFO) issued by the processor hits multiple tag fields in the L2 cache (the correct tag and the tag with the error) and the accessed data also has a correctable error, the processor will correctly log the multiple tag match error but will hang when attempting to execute the machine check exception handler.
- If a memory access receives a machine check error on both 64 byte halves of a 128-byte L2 cache sector, the IA32_MC0_STATUS register records this event as multiple errors, i.e., the valid error bit and the overflow error bit are both set indicating that a machine check error occurred while the results of a previous error were in the error-reporting bank. The IA32_MC1_STATUS register should also record this event as multiple errors but instead records this event as only one correctable error.
- The overflow bit should be set to indicate when more than one error has occurred. The overflow bit being set indicates that more than one error has occurred. Because of this erratum, if any further errors occur, the MCA overflow bit will not be updated, thereby incorrectly indicating only one error has been received.
- If an I/O instruction (IN, INS, REP INS, OUT, OUTS, or REP OUTS) is being executed, and if the data for this instruction becomes corrupted, the processor will signal a Machine Check Exception (MCE). If the instruction is directed at a device that is powered down, the processor may also receive an assertion of SMI#. Since MCEs have higher priority, the processor will call the MCE handler, and the SMI# assertion will remain pending. However, while attempting to execute the first instruction of the MCE handler, the SMI# will be recognized and the processor will attempt to execute the SMM handler. If the SMM handler is successfully completed, it will attempt to restart the I/O instruction, but will not have the correct machine state due to the call to the MCE handler. This can lead to failure of the restart and shutdown of the processor.

- If PWRGOOD is de-asserted during a RESET# assertion causing internal glitches, the MCA registers may latch invalid information.
- If RESET# is asserted, then de-asserted, and reasserted, before the processor has cleared the MCA registers, then the information in the MCA registers may not be reliable, regardless of the state or state transitions of PWRGOOD.
- If MCERR# is asserted by one processor and observed by another processor, the observing processor does not log the assertion of MCERR#. The Machine Check Exception (MCE) handler called upon assertion of MCERR# will not have any way to determine the cause of the MCE.
- The Overflow Error bit (bit 62) in the IA32_MC0_STATUS register indicates, when set, that a machine check error occurred while the results of a previous error were still in the error reporting bank (i.e. The Valid bit was set when the new error occurred). If an uncorrectable error is logged in the error-reporting bank and another error occurs, the overflow bit will not be set.
- The MCA Error Code field of the IA32_MC0_STATUS register gets written by a different mechanism than the rest of the register. For uncorrectable errors, the other fields in the IA32_MC0_STATUS register are only updated by the first error. Any further errors that are detected will update the MCA Error Code field without updating the rest of the register, thereby leaving the IA32_MC0_STATUS register with stale information.
- When a speculative load operation hits the L2 cache and receives a correctable error, the IA32_MC1_Status Register may be updated with incorrect information. The IA32_MC1_Status Register should not be updated for speculative loads.
- The processor should only log the address for L1 parity errors in the IA32_MC1_Status register if a valid address is available. If a valid address is not available, the Address Valid bit in the IA32_MC1_Status register should not be set. In instances where an L1 parity error occurs and the address is not available because the linear to physical address translation is not complete or an internal resource conflict has occurred, the Address Valid bit is incorrectly set.
- The processor may hang when an instruction code fetch receives a hard failure response from the system bus. This occurs because the bus control logic does not return data to the core, leaving the processor empty. IA32_MC0_STATUS MSR does indicate that a hard fail response occurred.
- The processor may hang when the following events occur and Machine Check Exceptions are enabled, CR4.MCE=1. A processor that has it's STPCLK# pin asserted will internally enter the Stop Grant State and finally issue a Stop Grant Acknowledge special cycle to the bus. If an uncorrectable error is generated during the Stop Grant process it is possible for the Stop Grant special cycle to be issued to the bus before the processor vectors to the machine check handler. Once the chipset receives its last Stop Grant special cycle it is allowed to ignore any bus activity from the processors. As a result, processor accesses to the machine check handler may not be acknowledged, resulting in a processor hang.

Implication: The processor is unable to correctly report and/or recover from certain errors.

Workaround: None identified at this time.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O12. Debug Mechanisms May Not Function as Expected

Problem: Certain debug mechanisms may not function as expected on the processor. The cases are as follows:

- When the following conditions occur: 1) An FLD instruction signals a stack overflow or underflow, 2) the FLD instruction splits a page-boundary or a 64 byte cache line boundary, 3) the instruction matches a Debug Register on the high page or cache line respectively, and 4) the FLD has a stack fault and a memory fault on a split access, the processor will only signal the stack fault and the debug exception will not be taken.
- When a data breakpoint is set on the ninth and/or tenth byte(s) of a floating point store using the Extended Real data type, and an unmasked floating point exception occurs on the store, the break point will not be captured.
- When any instruction has multiple debug register matches, and any one of those debug registers is enabled in DR7, all of the matches should be reported in DR6 when the processor goes to the debug handler. This is not true during a REP instruction. As an example, during execution of a REP MOVSW instruction the first iteration a load matches DR0 and DR2 and sets DR6 as FFFF0FF5h. On a subsequent iteration of the instruction, a load matches only DR0. The DR6 register is expected to still contain FFFF0FF5h, but the processor will update DR6 to FFFF0FF1h.
- A Data breakpoint that is set on a load to uncacheable memory may be ignored due to an internal segment register access conflict. In this case the system will continue to execute instructions, bypassing the intended breakpoint. Avoiding having instructions that access segment descriptor registers, e.g., LGDT, LIDT close to the UC load, and avoiding serialized instructions before the UC load will reduce the occurrence of this erratum.

Implication: Certain debug mechanisms do not function as expected on the processor.

Workaround: None identified.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O13. Processor May Timeout Waiting for a Device to Respond After 0.67 Seconds

Problem: The PCI 2.1 target initial latency specification allows two seconds for a device to respond during initialization-time. The processor may timeout after only approximately 0.67 seconds. When the processor times out it will hang with IERR# asserted. PCI devices that take longer than 0.67 seconds to initialize may not be initialized properly.

Implication: System may hang with IERR# asserted

Workaround: None identified at this time

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O14. Cascading of Performance Counters Does not Work Correctly when Forced Overflow is Enabled

Problem: The performance counters are organized into pairs. When the CASCADE bit of the Counter Configuration Control Register (CCCR) is set, a counter that overflows will continue to count in the other counter of the pair. The FORCE_OVF bit forces the counters to overflow on every non-zero increment. When the FORCE_OVF bit is set, the counter overflow bit will be set but the counter no longer cascades.

Implication: The performance counters do not cascade when the FORCE_OVF bit is set

Workaround: None identified at this time

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

O15. EMON Event Counting of x87 Loads May Not Work as Expected

Problem: If a performance counter is set to count x87 loads and floating-point exceptions are unmasked, the FPU Operand (Data) Pointer (FDP) may become corrupted.

Implication: When this erratum occurs, the FDP may become corrupted.

Workaround: This erratum will not occur with floating-point exceptions masked. If floating-point exceptions are unmasked, then performance counting of x87 loads should be disabled.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O16. Simultaneous Code Breakpoint and Uncorrectable Error Results in a Processor Hang

Problem: If an instruction fetch results in an uncorrectable error and there is also a debug breakpoint at this address, the processor will hang and the uncorrectable error will not be logged in the Machine Check registers.

Implication: When this erratum occurs the processor will livelock.

Workaround: None identified at this time

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O17. Software Controlled Clock Modulation using a 12.5% or 25% Duty Cycle May Cause the Processor to Hang

Problem: Processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL). The On-Demand Clock Modulation Duty Cycle is controlled by bits 3:1. If these bits are set to a duty cycle of 12.5% or 25%, the processor may hang while attempting to execute a floating-point instruction. In this failure, the last instruction pointer (LIP) is pointing to a floating-point instruction whose instruction bytes are in UC space and which takes a floating-point error exception. The processor continuously cycles attempting to fetch the bytes of the faulting floating-point instruction and those following it. This erratum is caused by interactions between the thermal control circuit and floating-point event handler.

Implication: When software controlled clock modulation is used with a duty cycle of 12.5% or 25% the processor will go into a sleep state from which it fails to return.

Workaround: Use a duty cycle other than 12.5% or 25%.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O18. Processor Samples Bus Frequency Power-On Configuration Pins at the Assertion of PWRGOOD

Problem: According to the *Intel® Xeon™ Processor MP Electrical, Mechanical, and Thermal Specifications (EMTS)*, the bus frequency-to-core ratio may be set by the power-on configuration option pins LINT[1:0], IGNNE#, and A20M#. The processor should only sample these pins on the active-to-inactive transition of RESET#, however, the processor is also sampling these pins on the inactive-to-active transition of PWRGOOD. The internal initialization done by the processor between the assertion of PWRGOOD and the de-assertion of RESET# may be affected if this ratio represents a high frequency at which the part will not properly function. This failure to initialize the processor properly may prevent the processor from coming out of reset or prevent some features such as the thermal control circuit from working properly.

Implication: The processor may fail to initialize properly if the frequency specified by the power-on configuration bits sampled at the assertion of PWRGOOD is too high for the processor to function correctly. On production parts and qualification samples, the frequency is internally limited so that this erratum should have no impact.

Workaround: No workaround is required for systems using qualification or production processors.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O19*.* PAT Index MSB May be Calculated Incorrectly

Problem: When Mode C or Mode B paging support is enabled and all of the following events occur:

- A page walk returns the Page Directory Entry (PDE) for a large page from memory.
- A subsequent page walk returns the Page Table Entry (PTE) for a 4k page from memory and the Page Attribute Table (PAT) upper index bit (bit 7) in this PTE is set to 1b.

It is possible that the PAT upper index bit in the PTE is incorrectly ignored and assumed to be 0b. The result is that the memory type in the PAT that should have come from the corresponding PAT index [4-7] incorrectly comes from PAT index [0-3].

Implication: If an operating system has programmed the PAT in an asymmetrical fashion i.e. PAT[0-3] is different from PAT[4-7] then an incorrect memory type may be used.

Workaround: None identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O20. System Bus Interrupt Messages Without Data Which Receive A Hard Failure Response may Hang the Processor

Problem: When a system bus agent (processor or chipset) issues an interrupt transaction without data onto the system bus, and the transaction receives a HardFailure response, a potential processor hang can occur. The processor, which generates an inter-processor interrupt (IPI) that receives HardFailure response, will still log the MCA error event cause as HardFalure, even if the APIC causes a hang. Other processors, which are true targets of the IPI, will also hang on hardfail-without-data, but will not record an MCA HardFailure event as cause. If a HardFailure response occurs on a system bus interrupt message with data, the APIC will complete the operation so as not to hang the processor.

Implications: The processor may hang after a HardFailure response to an IPI message thereby preventing the processor from saving the error state via an MCA handler.

Workaround: None Identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O21. Bus Invalidate Line Requests that Return Unexpected Data may Result in L1 Cache Corruption

Problem: When a Bus Invalidate Line (BIL) request receives unexpected data from a deferred reply, and a store operation write combines to the same address, there is a small window where the L1cache is corrupt, and loads can retire with this corrupted data. This erratum occurs in the following scenario:

- A Read-For-Ownership (RFO) transaction is issued by the processor and hits a line in shared state in the L2 cache.
- The RFO is then issued on the system bus as a 0 length Read-Invalidate (BIL), since it doesn't need data, just ownership of the cache line.
- This transaction is deferred by the chipset.
- At some later point, the chipset sends a deferred reply for this transaction with an implicit write-back response. For this erratum to occur, no snoop of this cache line can be issued between the BIL and the deferred reply.
- The processor issues a write-combining store to the same cache line while data is returning to the processor. This store straddles an 8-byte boundary.

Due to an internal boundary condition, a time window exists where the L1 cache contains corrupt data which could be accessed by a load.

Implication: The L1 cache may contain corrupted data. No known commercially available chipsets trigger the failure conditions.

Workaround: The chipset could issue a BIL (snoop) to the deferred processor to eliminate the failure conditions.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O22. Processor Flags #PF Instead of #AC on an Unlocked CMPXCHG8B Instruction

Problem: If a data page fault (#PF) and alignment check fault (#AC) both occur for an unlocked CMPXCHG8B instruction, then #PF will be flagged.

Implication: Software that depends #AC before #PF will be affected since #PF is flagged in this case.

Workaround: Remove the software's dependency on the fact that #AC has precedence over #PF. Alternately, if the page fault is due to a not present page, reload the page in the page fault handler and then restart the faulting instruction.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O23. Incorrect Data May Be Returned When Page Tables are Located in Write Combining (WC) Memory

Problem: If page directories and/or page tables are located in Write Combining (WC) memory, speculative loads to cacheable memory may complete with incorrect data.

Implication: Cacheable loads to memory mapped using page tables located in write combining memory may return incorrect data. Intel has not been able to reproduce this erratum with commercially available software.

Workarounds: Do not place page directories and/or page tables in WC memory.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O24. Multi-Processor Boot Protocol May Not Complete with an IOQ Depth of One

Problem: When the In-Order Queue (IOQ) depth is managed by the chipset to be one entry deep, the system may hang during the multi-processor boot protocol. This hang occurs when the chipset drives BNR# in such a way that the processors are continually throttled off the bus then released to access the bus in alternating cycles which never allows the multi-processor boot protocol to complete execution.

Implication: The system may hang during the multi-processor boot protocol.

Workaround: If the chipset drives BNR# in such a way that the processors are continually throttled off the bus then released to access the bus in alternating cycles, do not use In-Order Queue de-pipelining.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O25. Write Combining (WC) Load May Result In An Unintended Address On System Bus

Problem: When the processor performs a speculative write combining (WC) load, down the path of a mispredicted branch, and the address happens to match a valid UnCacheable (UC) address translation with the Data Translation Look-Aside Buffer, an unintended UnCacheable load operation may be sent out on the system bus.

Implication: When this erratum occurs, an unintended load may be sent on the system bus. Intel has only encountered this erratum during pre-silicon simulation.

Workaround: It is possible for the BIOS to contain a workaround for this erratum for some steppings of the processor.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O26. Processor Issues Inconsistent Transaction Size Attributes For Locked Operations

Problem: When the processor is in the Page Address Extension (PAE) mode and detects the need to set the Access and/or Dirty bits in the page directory or page table entries, the processor sends an 8 byte load lock onto the system bus. A subsequent 8 byte store unlock is expected, but instead a 4 byte store unlock occurs. Correct data is provided since only the lower bytes change, however external logic monitoring the data transfer may be expecting an 8-byte store unlock.

Implication: No known commercially available chipset are affected by this erratum.

Workaround: None identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O27. Multiple Accesses to the same S-State L2 Cache Line and ECC Error Combination May Result in Loss of Cache Coherence

Problem: When a Read For Ownership (RFO) cycle has a 64 bit address match with an outstanding read hit on a line in the L2 cache which is in the S-state AND that line contains an ECC error, the processor should recycle the RFO until the ECC error is handled. Due to this erratum, the processor does not recycle the RFO and attempt to service both the RFO and the read hit at the same time.

Implication: When this erratum occurs, cache may become incoherent.

Workaround: None identified at this time.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O28. IA32_MC0_ADDR and IA32_MC0_MISC Registers Will Contain Invalid or Stale Data Following a Data, Address, or Response Parity Error

Problem: If the processor experiences a data, address, or response parity error, the ADDRV and MISCV bits of the IA32_MC0_STATUS register are set, but the IA32_MC0_ADDR and IA32_MC0_MISC registers are not loaded with data regarding the error.

Implication: When this erratum occurs, the IA32_MC0_ADDR and IA32_MC0_MISC registers will contain invalid or stale data.

Workaround: Ignore any information in the IA32_MC0_ADDR and IA32_MC0_MISC registers after a data, address or response parity error.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O29. Instruction Pointer Stored on Stack May Become Invalid

Problem: Due to an internal boundary condition which may exist on a Hyper-Threading Technology enabled Intel® Xeon™ processor MP, the following sequence of events must occur:

a) One logical processor executes the WRMSR instruction with incorrect data causing a general protection fault

b) Simultaneously an event that requires micro-architectural synchronization among the two logical processors occurs on the second logical processor may cause an invalid instruction pointer to be stored on the ring 0 stack during the transition to GP fault handler on the first logical processor.

Implication: The instruction pointer stored on the stack may be invalid, potentially causing errors during execution of or return from the GP fault handler.

Workaround: It is possible for BIOS to contain a workaround this issue. For Hyper-Threading Technology enabled processors; insure all WRMSR instructions do not generate GP faults due to incorrect data.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

.

O30. When the Processor is in the System Management Mode (SMM), Debug Registers may be Fully Writeable

Problem: When in System Management Mode (SMM), the processor executes code and stores data in the SMRAM space. When the processor is in this mode and writes are made to DR6 and DR7, the processor should block writes to the reserved bit locations. Due to this erratum, the processor may not block these writes. This may result in invalid data in the reserved bit locations.

Implication: Reserved bit locations within DR6 and DR7 may become invalid.

Workaround: Software may perform a read/modify/write when writing to DR6 and DR7 to ensure that the values in the reserved bits are maintained.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O31. Associated Counting Logic must be Configured when Using Event Selection Control (ESCR) MSR

Problem - ESCR MSRs allow software to select specific events to be counted, with each ESCR usually associated with a pair of performance counters. ESCRs may also be used to qualify the detection of at-retirement events that support precise-event-based sampling (PEBS). A number of performance metrics that support PEBS require a 2^{nd} ESCR to tag uops for the qualification of at-retirement events. (The first ESCR is required to program the at-retirement event.) Counting is enabled via counter configuration control registers (CCCR) while the event count is read from one of the associated counters. When counting logic is configured for the subset of at-retirement events that require a 2^{nd} ESCR to tag uops, at least one of the CCCRs in the same group of the 2^{nd} ESCR must be enabled.

Implication - If no CCCR/counter is enabled in a given group, the ESCR in that group that is programmed for tagging uops will have no effect. Hence a subset of performance metrics that require a 2^{nd} ESCR for tagging uops may result in 0 count.

Workaround - Ensure that at least one CCCR/counter in the same group as the tagging ESCR is enabled for those performance metrics that require two ESCRs and tagging uops for at-retirement counting.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O32. Livelock may Occur when Bus Parking is Disabled

Problem: A livelock may occur when processor bus parking is disabled, and when (1) the processor is the symmetric owner of the bus with one internal request pending, and (2) the processor observes the assertion of BPRI#. The processor assumes that it will assert ADS# and deasserts BREQ without issuing its pending request. Assertion of BPRI# coincident with the arbitration phase of the same processor that still has only one outstanding internal request will result in that processor being livelocked. Any change to the regular pattern of BPRI# assertion noted above or the arrival of a second internal transaction will release the processor from the livelock condition.

Implication: This erratum may result in a livelock.

Workaround: This erratum can be avoided by enabling bus parking.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O33. CPUID Function 2 May Return Incorrect Cache Size Information

Problem: When a Hyper-Threading Technology-enabled processor executes a CPUID instruction with function 2 (02 in the EAX register), the processor may return incorrect/invalid cache descriptors in the EDX register. Code must be executing on both logical processors to encounter this erratum.

Implication: When this erratum occurs the data returned to the EDX register may be inaccurate/invalid.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O34. CR2 May Be Incorrect or an Incorrect Page Fault Error Code May Be Pushed onto Stack After Execution of an LSS Instruction

Problem: Under certain timing conditions, the internal load of the selector portion of the LSS instruction may complete with potentially incorrect speculative data before the load of the offset portion of the address completes. The incorrect data is corrected before the completion of the LSS instruction but the value of CR2 and the error code pushed on the stack are reflective of the speculative state. Intel has not observed this erratum with commercially available software.

Implication: When this erratum occurs, the contents of CR2 may be off by two, or an incorrect page fault error code may be pushed onto the stack.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O35. Hyper-Threading Technology Enabled Processors May Hang in the Presence of Extensive Self-Modifying Code

Problem: For multi-processor platforms, in which Hyper-Threading Technology enabled processors are executing extensive self modifying code, and branch trace messages are enabled on at least one logical processor, the system may hang. In this scenario, a processor executing within 1K of code being written to by another processor may attempt to end this flow, thereby resulting in a hang.

Implication: When this erratum occurs the system will hang.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O36. Global Bit Incorrectly Set for Secondary Logical Processors in ITLB

Problem: Due to a boundary condition in the translation look-aside buffer logic, the global bit information in the TLB entry for a mapping belonging to the first logical processor can overwrite the global bit information for a mapping belonging to the second logical processor. This occurs in the following scenario:

- The first logical processor misses the ITLB resulting in a page walk
- The second logical processor also misses the ITLB and generates a page walk

In certain timing scenarios within the processor, the leftover global bit information from the first logical processor may overwrite the second logical processor.

Implication: When this erratum occurs, if the page global bit for the second logical processor is overwritten with a 0b, this will result in performance degradation for the first logical processor. If the page global bit is incorrectly changed from a 0 to 1, this erratum may result in software failures.

Workaround: It is possible for BIOS code to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O37. Hardware Prefetcher may Cause Stale Data to be Loaded into the Processor Caches

Problem: The processor may use stale data from the cache while the Hardware Prefetcher is enabled. The conditions of this erratum are as follows:
A cacheline is stored in the L3 cache in shared state while its adjacent sector is in modified state. The same cacheline and its adjacent sector reside in the L2 cache in the shared and invalid state respectively. The cacheline and its adjacent sector are being evicted from the L3 cache at the same time that a prefetch RFO is issued to this address. A boundary condition exists in the bus logic where the prefetch may be issued on the system bus before the modified data in the L3 is written back to main memory. Consequently the RFO gets stale data for the adjacent sector from main memory and fills the cache with this stale data.

Implication: The processor may use stale data from the cache.

Workaround: Disable the Hardware Prefetcher by setting bit 9 in register IA32_MISC_ENABLE - MSR Address 01A0h via the BIOS.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

O38. System May Hang if a Fatal Cache Error Causes Bus Write Line (BWL) Transaction to Occur to the Same Cache Line Address as an Outstanding Bus Read Line (BRL) or Bus Read-Invalidate Line (BRIL)

Problem: A processor internal cache fatal data ECC error may cause the processor to issue a BWL transaction to the same cache line address as an outstanding BRL or BRIL. As it is not typical behavior for a single processor to have a BWL and a BRL/BRIL concurrently outstanding to the same address, this may represent an unexpected scenario to system logic within the chipset.

Implication: The processor may not be able to fully execute the machine check handler in response to the fatal cache error if system logic does not ensure forward progress on the system bus under this scenario.

Workaround: System logic should ensure completion of the outstanding transactions. Note that during recovery from a fatal data ECC error, memory image coherency of the BWL with respect to BRL/BRIL transactions is not important. Forward progress is the primary requirement.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O39. Re-mapping the APIC Base Address to a Value Less Than or Equal to 0xDC001000 may Cause IO and Special Cycle Failure

Problem: Re-mapping the APIC base address from its default can cause conflicts with either I/O or special cycle bus transactions.

Implication: Either I/O or special cycle bus transactions can be redirected to the APIC, instead of appearing on the front-side bus.

Workaround: Use any APIC base addresses above 0xDC001000 as the relocation address.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O40. Erroneous Machine Check Error Reported

Problem: An erroneous multi-bit ECC Machine Check Error may occur on Hyper-Threading Technology enabled processors when both logical processors are in the halt state. In this state, the clocks inside the processor will be shut off. There is a boundary case where a speculative page walk could be occurring while the clocks are shut off. This page walk continues after the clocks are turned back on. If the clocks are off when the page walk is in a specific pipe stage in the machine, an erroneous L2 tag ECC error may be observed.

Implication: Due to this erratum, an erroneous multi-bit error may be reported in the Machine Check registers when Machine Check is enabled. There is no known impact when Machine Check is disabled. There have been no observances of data corruption caused by this issue.

Workaround: It is possible for BIOS to contain a workaround for this issue.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O41. Processor Does not Flag #GP on Non-zero Write to Certain MSRs

Problem: When a non-zero write occurs to the upper 32 bits of IA32_CR_SYSENTER_EIP or IA32_CR_SYSENTER_ESP, the processor should indicate a general protection fault by flagging #GP. Due to this erratum, the processor does not flag #GP.

Implication: The processor unexpectedly does not flag #GP on a non-zero write to the upper 32 bits of IA32_CR_SYSENTER_EIP or IA32_CR_SYSENTER_ESP. No known commercially available operating system has been identified to be affected by this erratum.

Workaround: None identified.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

O42. Counting Both L2 and L3 Cache Reference Events may Result in Undercount

Problem: The processor's counting logic combines events from misaligned pipestages. Thus, if two requests are sent referencing L2 and L3 at the same time, the EMON logic may only log them as one event.

Implication: This may result in undercounting of cache reference events.

Workaround: Restrict to counting either L2 or L3 events, but not both at the same time on a single ESCR.

O43. Simultaneous Assertion of A20M# and INIT# may Result in Incorrect Data Fetch

Problem: If A20M# and INIT# are simultaneously asserted by software, followed by a data access to the 0xFFFFFXXX memory region, with A20M# still asserted, incorrect data will be accessed. With A20M# asserted, an access to 0xFFFFFXXX should result in a load from physical address 0xFFEFFXXX. However, in the case of A20M# and INIT# being asserted together, the data load will actually be from the physical address 0xFFFFFXXX. Code accesses are not effected by this erratum.

Implication: Processor may fetch incorrect data, resulting in BIOS failure.

Workaround: Deasserting and reasserting A20M# prior to the data access will workaround this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section

DOCUMENTATION CHANGES

The Documentation Changes listed in this section apply to the following documents:

- *Intel® Xeon™ Processor MP* datasheet (Order Number 29074001)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

All Documentation Changes will be incorporated into a future version of the appropriate Intel® Xeon™ processor documentation.

O3. ISR Must Re-enable CCCR After Each PEBS Overflow

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Section 15.5.7.4, under title "WRITING THE DS INTERRUPT SERVICE ROUTINE" a new bullet is added at the end of the section.

- The ISR must re-enable the CCCR's ENABLE bit if it is servicing an overflow PMI due to PEBS.

O4. Sequence to Programming Performance Counters

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Section 15.9.6, under title "Programming the Performance Counters for Non-Retirement Events", currently states:

2. Select the performance counter to count the events and an associated ESCR to select the events to be counted.

It should state:

2. For each event, select an ESCR that supports the event using the values in the ESCR Restrictions row in Table A-1.

3. Match the CCCR Select value and ESCR name in Table A-1 to the values listed ESCR Name and ESCR No. columns in Table 15-4, to select a CCCR and performance counter.

O5. 0x2B MSR Definition

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Appendix B, under Table B1, MSR_EBC_SOFT_POWERON, bit 4, currently states:

Initiator MCERR# Disable. (R/W) Set to disable MCERR# driving for initiator bus requests (default); clear to disable.

It should state:

Initiator MCERR# Disable. (R/W) Set to disable MCERR# driving for initiator bus requests (default); clear to enable.

O6. ESI and EDI Alignment For Fast String Moves

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Chapter 7, Section 7.2.3, first bullet of the third paragraph currently states:

> Source and destination addresses must be 8-byte aligned.

It should state:

> EDI and ESI must be 8-byte aligned for the Pentium III processor. EDI must be 8-byte aligned for the Pentium 4 processor.

O7. BUS_UTILIZATION_DUE_TO_PROCESSOR_ACTIVITY Event Number Correction

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* in Table A-8 under Mnemonic Event Name "BUS_UTILIZATION_DUE _TO_PROCESSOR_ACTIVITY (Counter 0)", Event Number currently states:

> 2DH

It should state:

> 2EH

O8. Complement flag, bit 19

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide,* Chapter 15,Section 15.9.3, CCCR MSRs, last sentence of the "Complement flag, bit 19" paragraph on page 15-30 currently states:

> The compare flag is not active unless the compare flag is set.

It should state:

> The complement flag is not active unless the compare flag is set.

O9. BSF and BSR Incorrectly Documented in Vol 2 Appendix B

The *Intel Architecture Software Developer's Manual, Vol* 2*: Instruction Set Reference* Appendix B-7, Table B-10 states:

BSF - Bit Scan Forward
register1, register2 0000 1111 : 1011 1100 : 11 reg2 reg1
memory, register 0000 1111 : 1011 1100 : mod reg r/m
BSR - Bit Scan Reverse
register1, register2 0000 1111 : 1011 1101 : 11 reg2 reg1
memory, register 0000 1111 : 1011 1101 : mod reg r/m

It should state:

BSF - Bit Scan Forward
register1, register2 0000 1111 : 1011 1100 : 11 reg1 reg2
memory, register 0000 1111 : 1011 1100 : mod reg r/m
BSR - Bit Scan Reverse
register1, register2 0000 1111 : 1011 1101 : 11 reg1 reg2
memory, register 0000 1111 : 1011 1101 : mod reg r/m

O10. Performance Counter MSRs (MSR_IQ_COUNTER)

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Section 15.9, Counter No.12, 13, and 16 of Table 15-4, it currently states:

MSR_IQ_COUNTER0	12	30CH	MSR_IQ_CCCR0	36CH	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	3	3BCH
					MSR_SSU_ESCR0	2	3BEH
					MSR_ALF_ESCR0	1	3CAH
MSR_IQ_COUNTER1	13	30DH	MSR_IQ_CCCR1	36DH	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	3	3BCH
					MSR_SSU_ESCR0	2	3BEH
					MSR_ALF_ESCR0	1	3CAH
MSR_IQ_COUNTER4	16	310H	MSR_IQ_CCCR4	370H	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	3	3BCH
					MSR_SSU_ESCR0	2	3BEH
					MSR_ALF_ESCR0	1	3CAH

It should state:

MSR_IQ_COUNTER0	12	30CH	MSR_IQ_CCCR0	36CH	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	2	3BCH
					MSR_SSU_ESCR0	3	3BEH
					MSR_ALF_ESCR0	1	3CAH
MSR_IQ_COUNTER1	13	30DH	MSR_IQ_CCCR1	36DH	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	2	3BCH
					MSR_SSU_ESCR0	3	3BEH
					MSR_ALF_ESCR0	1	3CAH
MSR_IQ_COUNTER4	16	310H	MSR_IQ_CCCR4	370H	MSR_CRU_ESCR0	4	3B8H
					MSR_CRU_ESCR2	5	3CCH
					MSR_CRU_ESCR4	6	3E0H
					MSR_IQ_ESCR0	0	3BAH
					MSR_RAT_ESCR0	2	3BCH
					MSR_SSU_ESCR0	3	3BEH
					MSR_ALF_ESCR0	1	3CAH

O11. Tagging Mechanism for Replay_Event

The *Intel Architecture Software Developer's Manual, Vol 3: System Programming Guide* Section 15.9.7.4., in the 3rd paragraph, currently states:

The Table A-5 lists the metrics that are support the replay tagging mechanism and the at-retire-ment events that use the replay tagging mechanism, and specifies how the appropriate MSRs need to be configured. Each of these metrics requires that the Replay_Event (see Table A-2) be used to count the tagged µops.

It should state:

The Table A-5 lists the metrics that are support the replay tagging mechanism and the at-retire-ment events that use the replay tagging mechanism, and specifies how the appropriate MSRs need to be configured. The replay tags defined in Table A-5 also enable Precise Event-Based Sampling (PEBS, see Section 15.9.8). Each of these replay tags can also be used in normal sampling by not setting Bit 24 nor Bit 25 in IA32_PEBS_ENABLE_MSR. Each of these metrics requires that the Replay_Event (see Table A-2) be used to count the tagged µops.

O12. Update to Table B-2, MSRs in the P6 Family Processors

In Table B-2 in the *IA-32 Intel Architecture Software Developer's Manual, Volume 3: System Programming Guide* two changes will be made. For the MC0_STATUS register, bit 60 row, the Bit Description will be revised to read, "MC_STATUS_EN. (Note: For MC0_STATUS only, this bit is hard-coded to 1.)" For the MC4_STATUS register row the Bit Description will be revised to read, "Bit definitions same as MC0_STATUS, except bits 0, 4, 57, and 61 are hard-coded to 1."

SPECIFICATION CHANGES

The Specification Changes listed in this section apply to the following documents:

- *Intel® Xeon™ Processor MP* datasheet (Order Number 29074001)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

All Specification Changes will be incorporated into a future version of the appropriate Intel® Xeon™ processor documentation.

There are no specification changes to report.